DETAILED ACTION

1. Claims 1, 3 – 8, 10, 12 - 23 are pending for examination. This office action is in response to amendment filed 04/06/2009.

Double Patenting

2. The nonstatutory double patenting rejection is based on a judicially created doctrine grounded in public policy (a policy reflected in the statute) so as to prevent the unjustified or improper timewise extension of the "right to exclude" granted by a patent and to prevent possible harassment by multiple assignees. A nonstatutory obviousness-type double patenting rejection is appropriate where the conflicting claims are not identical, but at least one examined application claim is not patentably distinct from the reference claim(s) because the examined application claim is either anticipated by, or would have been obvious over, the reference claim(s). See, e.g., *In re Berg*, 140 F.3d 1428, 46 USPQ2d 1226 (Fed. Cir. 1998); *In re Goodman*, 11 F.3d 1046, 29 USPQ2d 2010 (Fed. Cir. 1993); *In re Longi*, 759 F.2d 887, 225 USPQ 645 (Fed. Cir. 1985); *In re Van Ornum*, 686 F.2d 937, 214 USPQ 761 (CCPA 1982); *In re Vogel*, 422 F.2d 438, 164 USPQ 619 (CCPA 1970); and *In re Thorington*, 418 F.2d 528, 163 USPQ 644 (CCPA 1969).

A timely filed terminal disclaimer in compliance with 37 CFR 1.321(c) or 1.321(d) may be used to overcome an actual or provisional rejection based on a nonstatutory double patenting ground provided the conflicting application or patent either is shown to be commonly owned with this application, or claims an invention made as a result of activities undertaken within the scope of a joint research agreement.

Effective January 1, 1994, a registered attorney or agent of record may sign a terminal disclaimer. A terminal disclaimer signed by the assignee must fully comply with 37 CFR 3.73(b).

3. Claims 1, 3 – 8, 10, 12 - 23 are provisionally rejected on the ground of nonstatutory obviousness-type double patenting as being unpatentable over claims 1, 3 – 8, 10 of copending Application No. 11/862,057 (refers as 057) in view of Nasika "Transparent Migration of Distributed Communicating Processes". Although the

conflicting claims are not identical, they are not patentably distinct from each other because both computer systems comprise substantially the same elements.

Both 057 and this application teach:

a base operating system (OS) installed to the computing machine, the base OS having a base OS file system and a base OS registry;

at least one virtual OS environment within the base OS, the virtual OS environment having a virtual OS file system and a virtual OS registry which are independent of the base OS file system and the base OS registry;

redirecting, attempts of the application to access the base OS file system and the base OS registry to the virtual OS.

057 does not teach:

injecting a dynamic link library (DLL) into an application running under the virtual OS environment, and the redirecting is via the DLL.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the teachings of Nasika and 059 because because injecting a DLL into an application would manage the execution of the application and endows it with features such as virtualization and mobility (Nasika; abstract).

This is a <u>provisional</u> obviousness-type double patenting rejection.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 1, 3 - 8, 10, 12 – 14, 16 – 18, 20 - 23 are rejected under 35 U.S.C. 103(a) as being unpatentable over Calder, US pub. no. 2002/0092003 in view of Nasika, "Transparent Migration of Distributed Communicating Processes".

6. Calder reference was cited in previous office action.

7. **As to claim 1,** Calder teaches computing machine having a computing architecture, comprising:

a base operating system (OS) installed to the computing machine (operating system of the client computer 140, figures 3 and 4 and associated text, especially [0086, 0094, 0096]), the base OS having a base OS file system (files system 360, figures 3 and 4 and associated text) and a base OS registry (registry 350, figures 3 and 4 and associated text);

at least one virtual OS environment within the base OS, the virtual OS environment having a virtual OS file system and a virtual OS registry which are independent of the base OS file system (virtualized files system 435 of figure 4) and the base OS registry (virtualized registry 430);

redirecting, wherein the computing machine is configured such that attempts of the application to access the base OS file system and the base OS registry to the virtual OS file system and the virtual OS registry (all DLL routines that are intercepted are redirected, [0086, 0096, 0103, 0104, 0119]).

Calder does not explicitly teach the step of injecting a dynamic link library (DLL) into an application running under the virtual OS environment, and redirecting via an dll.

Nasika teaches the step of injecting a dynamic link library (DLL) into an application running under the virtual OS environment (injecting a wrapper DLL into an application at runtime, abstract, section 2), and the redirecting is via the DLL (the application passes a virtual handle to a system call, the virtual operating system (vos) intercept that calls and passes the…call, section 1.1 and 2, and figure 2).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the teachings of Calder and Nasika because injecting a DLL into an application would manage the execution of the application and endows it with features such as virtualization and mobility (Nasika; abstract).

8. **As to claim 3**, Calder teaches at least one application (program, p. 496 col. 2 and 3) running under the virtual OS environment, and wherein the application shares one or more of the following with the base OS: networking information (virtualized

network request, figure 36 and associated text), user login rights, services, hardware information, and clipboard information.

9. **As to claim 4**, Calder teaches a change made in the OS, wherein the change does not affect the main OS or any other virtual OS environment ([0086, 0103, 0104, 0119]).

10. **As to claim 5**, Calder teaches wherein each virtual OS environment contains a group of installed applications (registry 350, files system 360) that run independently of each another.

11. **As to claim 6**, Calder teaches one or more applications running under the base OS and each virtual OS environment, and wherein all of the applications run on a single OS desktop (client computer 140, figure 1 and associated text, [0077]).

12. **As to claim 7**, see rejection for claim 4 above.

13. **As to claim 8**, this is the method claim of claim 1. See rejection for claim 1 above.

14. **As to claim 10**, Calder teaches altering one or more application programming interfaces that access the base OS file system and registry directly and indirectly so as to redirect these accesses into the appropriate virtual file system and registry ([0119]).

15. **As to claims 12 - 13**, Calder teaches creating a copy of base OS file system and registry in the virtual OS environment file system and registry locations, and an application running under the virtual OS environment is executed using the copy in the virtual OS environment file system ([0119]).

16. **As to claim 14**, Calder teaches setting a predetermined directory such that an application running under the predetermined directory will be redirected to the virtual OS environment (sandbox directory, 0097, 0010, 0133- 0135).

17. **As to claim 16**, Calder teaches a method with a base operating system (OS) installed to the computing machine (operating system of the client computer 140, figures 3 and 4 and associated text, especially [0086, 0094, 0096]), the base OS having a base OS file system (files system 360, figures 3 and 4 and associated text) and a base OS registry (registry 350, figures 3 and 4 and associated text), the method comprising the steps of:

Creating a virtual OS environment within the base OS, the virtual OS environment having at least one of:

virtual OS file system which is independent of the base OS file system (virtualized files system 435 of figure 4);

the base OS registry (virtualized registry 430);

redirecting, an attempt by the application to access the base OS to at least one of:

the virtual OS file system and the virtual OS registry (all DLL routines that are intercepted are redirected to the virtualized system, [0086, 0103, 0104, 0119]).

Calder does not explicitly teach the step of injecting a dynamic link library (DLL) into an application running under the virtual OS environment, and the redirecting is via the DLL

Nasika teaches the step of injecting a dynamic link library (DLL) into an application running under the virtual OS environment (injecting a wrapper DLL into an application at runtime, abstract, section 2); and the redirecting is via the DLL (the application passes a virtual handle to a system call, the virtual operating system (vos) intercept that calls and passes the…call, section 1.1 and 2, and figure 2).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the teachings of Calder and Nasika because injecting a

DLL into an application would manage the execution of the application and endows it with features such as virtualization and mobility (Nasika; abstract).

18. **As to claim 17,** Calder modified by Nasika teaches determining that the application should be run under the virtual OS environment instead of the base OS, wherein the DLL performs the determining; scanning a function import table of the application (Calder; import table, [0094, 0096]), wherein redirecting the attempt by the application comprises redirecting, via the DLL, file system and registry calls from the application to functions within the injected DLL, wherein the DLL performs the scanning (Calder; scanned, [0094]).

19. **As to claim 18**, Nasika teaches determining that the application should be run under the virtual OS environment instead of the base OS, wherein the DLL performs the determining; calling a function of the injected DLL rather than calling an API of the base OS, wherein the DLL performs the calling; modifying at least one parameter from a calling function of the application (Nasika; change the mapping between physical handle and physical handle, section 1.1) to direct the at least one parameter to a location of the virtual OS environment, wherein the DLL performs the modifying.

20. **As to claim 20**, Calder teaches the attempt to access the base OS comprises an attempt to install at least one file in the base OS file system (installed, [0111, 0149]).

21. As to claim 21, Calder teaches the attempt to install the at least one file in the base OS file system comprises an attempt to reboot the computer (the computer continues executing, [0116, 0119]).

22. **As to claim 22**, Calder teaches shutting down the application running under the at least one virtual OS environment; after shutting down the application (shutdown, 0126, 0155]), installing (replaced the file in virtualized system, [0110, 0111, 0215]) the at least one file in the virtual OS file system of the at least one virtual OS environment; after installing the at least one file, executing the application under the at least one virtual OS environment (execution environment, figure2 and associated text, [00116, 0119]).

23. **As to claim 23**, Calder teaches wherein: shutting down the application running under the at least one virtual OS environment and installing the at least one file in the virtual OS file system of the at least one virtual OS environment are conducted while the base OS is running ([0119]).

24. Claim 15 is rejected under 35 U.S.C. 103(a) as being unpatentable over Calder, US pub. no. 2002/0092003 in view of Nasika, "Transparent Migration of Distributed Communicating Processes", and further in view of Bersson, US patent no. 6,081,897.

25. **As to claim 15**, Calder and Nasika do not explicitly teach the step of wherein the predetermined directory is a CD/DVD drive in the base OS file system.

Bersson teaches the predetermined directory is in drive in the base OS file system (second driver having a first routine to monitor ..... CD track, col. 2 lines 1 – 10).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Calder, Nasika, and Bersson because intercepting command at second driver would be able to monitor the input CD of each copyrighted CD track to be copied (Bersson; col. 2 lines 1 – 10).

Response to Arguments

26. Applicant's arguments have been considered but are moot in view of the new ground(s) of rejection.

27. During the prosecution of this application, examiner found that the continuation of this application has been filed after this application. Therefore, this double patenting rejection was not rejected before.

Allowable Subject Matter

28. Claim 19 is objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

Conclusion

29. The prior art made of record but not relied upon request is considered to be pertinent to applicant's disclosure.

Leverenz, US patent no. 6,754,889, demonstrating a method of enabling injection of non-native code into a Java environment.

30. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL.** See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to PHUONG N. HOANG whose telephone number is (571)272-3763. The examiner can normally be reached on Monday - Friday 9:00 am to 5:30 pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Hyunh S. Sough can be reached on 571-272-6799. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Li B. Zhen/
Primary Examiner, Art Unit 2194

/P. N. H./
Examiner, Art Unit 2194